COOLEY LLP
BOBBY GHAJAR (198719)
(bghajar@cooley.com)
TERESA MICHAUD (296329)
(tmichaud@cooley.com)
COLETTE GHAZARIAN (322235)
(cghazarian@cooley.com)
1333 2nd Street, Suite 400
Santa Monica, California 90401
Telephone: (310) 883-6400

MARK WEINSTEIN (193043)
(mweinstein@cooley.com)
KATHLEEN HARTNETT (314267)
(khartnett@cooley.com)
JUDD LAUTER (290945)
(jlauter@cooley.com)
ELIZABETH L. STAMESHKIN (260865)
(lstameshkin@cooley.com)
3175 Hanover Street
Palo Alto, CA 94304-1130
Telephone: (650) 843-5000

CLEARY GOTTLIEB STEEN & HAMILTON LLP
ANGELA L. DUNNING (212047)
(adunning@cgsh.com)
1841 Page Mill Road, Suite 250
Palo Alto, CA 94304
Telephone: (650) 815-4131

[Full Listing on Signature Page]

Counsel for Defendant Meta Platforms, Inc.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

RICHARD KADREY, *et al.*,

Individual and Representative Plaintiffs,

v.

META PLATFORMS, INC., a Delaware corporation;

Defendant.

Case No. 3:23-cv-03417-VC-TSH

DECLARATION OF BARBARA FREDERIKSEN-CROSS IN SUPPORT OF META'S OPPOSITION TO PLAINTIFFS' MOTION FOR PARTIAL SUMMARY JUDGMENT

I, Barabara Frederiksen-Cross, declare as follows:

1. I am over the age of 18 and am competent to make this declaration. I have been engaged by Meta Platforms, Inc. ("Meta") as a technical expert to provide my opinion regarding certain aspects of the downloading of certain datasets used in the development or training of large language models ("LLMs"), in particular the use of BitTorrent to download certain datasets. I submitted a rebuttal expert report in this matter on February 10, 2025 ("**Rebuttal Report**"), in response to an opening expert report of Jonathan Krein, Ph.D., served on January 10, 2025.

2. I understand that portions of my Rebuttal Report were attached as Exhibit 72 to the Declaration of Maxwell V. Pritt (Dkt. 472). All of the statements in my Rebuttal Report are accurate to the best of my knowledge, information, and belief. Except as otherwise stated herein, I make this declaration based on my personal knowledge and professional expertise, as well as Meta documents and source code produced in this action as well as research conducted by me or under my direction.

3. My Rebuttal Report addressed allegations made by Dr. Krein and Plaintiffs that Meta, during the download of certain datasets via BitTorrent, distributed Plaintiffs' works to third parties by "seeding" those works, which refers to when a computer on the BitTorrent network is able to upload pieces of a torrent file after the download has completed.[1] I further explained why, for various reasons summarized below, it was highly unlikely that Meta seeded Plaintiffs' works. The Krein report did not articulate any other theory beyond "seeding" by which Meta allegedly uploaded Plaintiffs' works in connection with Meta's dataset download process.

4. On February 26, 2025, Plaintiffs served a "Rebuttal Report of David R. Choffnes, Ph.D." ("**Choffnes Report**"), styled as a rebuttal to my Rebuttal Report. The Choffnes Report argues that, separate from "seeding," Meta likely uploaded at least one piece of Plaintiffs' works to another peer during the "leeching" phase, which refers to the time period during which a computer has not yet fully downloaded a torrent file. (Choffnes Report, ¶ 19.) Because this theory was not articulated in the Krein report, I have not had an opportunity to address allegations of

[1] Frederiksen-Cross Rebuttal Report, ¶¶ 51, 66-67, 89-90.

leeching, let alone the specific analysis and opinions set forth in the Choffnes Report. I have thus been asked to assess and comment on the Choffnes Report and respond to the assertion that Meta shared any of the Plaintiffs' works during the leeching phase of the dataset download process. As explained below, many of the factors identified in my Rebuttal Report that made seeding of Plaintiffs' works unlikely, also apply to the Choffnes Report's assertions regarding leeching, with additional factors that I describe below.

Professional Background and Qualifications

5. I am the Director of Litigation Services for JurisLogic, LLC ("JurisLogic"), an Oregon corporation that provides consulting services to computer hardware and software manufacturers and computer-related technical assistance to the legal profession in the United States, Canada, Japan, China, Europe, and the UAE. My experience includes software design, programming, project management, capacity planning, performance tuning, problem diagnosis, and administration of hardware, operating systems, application software, and database management systems. I have more than fifty (50) years of experience as a software developer and consultant, including the analysis of computer-based data, software development of web-based systems, and software development for secure online data systems used by banks, insurance companies, hospitals, and telecommunication providers.

6. I have experience in the design, development, and analysis of computer software, and I have previously provided both trial and deposition testimony as an expert for matters in federal and state courts, authored a number of papers, and delivered lectures on technology to the legal profession. I also have substantial prior experience with the analysis of peer-to-peer ("P2P") file distribution networks, including both the analysis of BitTorrent-related source code and testing related to BitTorrent's operation. This experience includes tests in which I created torrent files, used BitTorrent to upload and download files, used BitTorrent to transfer files over computer networks, and captured and analyzed BitTorrent network traffic. I have also studied both publicly available BitTorrent source code (including libtorrent and other implementations), as well as source code for proprietary file distribution software that uses the BitTorrent protocol, and proprietary systems used to monitor BitTorrent activity.

7. My background and experience also include the design, implementation, and ongoing administration of databases and multi-dimensional data aggregation systems, including data extraction, transfer, and load operations used for data analysis platforms. My experience also includes programming for embedded and robotic systems. I also have experience with computer and network capacity management, storage management, and disaster recovery planning and testing. I have previously qualified as an expert in federal and state courts to testify about the operation of computer software and computer systems, including for matters that involve software development disputes, failed software systems, and patent, copyright and trade secret disputes. I have also previously testified as an expert in several litigation matters that involved BitTorrent technologies. My work in these other matters included analysis and testimony relating to systems that monitor P2P file distribution systems and send notices to Internet Service Providers ("ISPs") based on the activity they detect.

8. I am a member of the Institute of Electrical and Electronics Engineers ("IEEE") and the Association for Computing Machinery ("ACM"). I am a salaried employee and one of the principals of JurisLogic. JurisLogic is compensated at the rate of $595 an hour for my work in this case. My compensation does not in any way depend on the substance of my opinions or the outcome of this or any other case.

Background on BitTorrent and Torrents

9. My previous Rebuttal Report provided an extensive background discussion about the basics of accessing files over the Internet, as well as the technologies that underlie peer-to-peer (P2P) network protocols including BitTorrent.[2] For the convenience of the Court, I will provide a concise summary of that information here.

10. The majority of the interaction for data on the internet flows through what is commonly known as a ***client-server model***, in which (a) the client computer requests data for download and (b) a centralized server responds to the request and provides the requested data. One of the primary limitations of the client-server model is the dependency on a central server or system for delivery of required data to clients. When a significant number of clients request the same

[2] Frederiksen-Cross Rebuttal Report, ¶¶ 21-73.

resource, the central server or system can become overloaded, causing slowdowns and failures for some clients. Another related drawback is that, because a central server or system is solely responsible for retrieving and transmitting requested data, it provides a "single point of failure" such that if it fails the data may become inaccessible to clients. Moreover, any communication within the client-server model is restricted by the bandwidth available to the central server or system. This can result in the transfer of very big files taking a large amount of time – particularly if there are many clients requesting the data – as the data must be individually transmitted by the central server or system within the limited bandwidth available to it.

11. Peer-to-peer (P2P) networks were designed to address many of these issues. They operate in a similar fashion to the client-server model but with one key difference – instead of a centralized server or system, in a P2P network each computer in the network may act as both a client and a server. Computers within a P2P network can thus operate as ***peers*** that can request specific information from other peers who possess it, or can provide the required information to other peers that are specifically requesting it. This differs from the client-server model in which only the centralized server or system holds the information that can be transmitted to connected clients. As a result, whereas the bandwidth in a client-server network is restricted by the bandwidth of the central server or system, the nodes in a P2P network act as both clients and servers, so each additional node can contribute to the total bandwidth of the network. Thus, P2P networks are inherently decentralized networks.

12. When using the BitTorrent protocol, any large file is first converted to smaller ***pieces***. In practice, when a peer using BitTorrent requests specific information (e.g. a request for a particular file), multiple peers that have that information and are participating in the network may respond by providing a portion of the requested file. Since the pieces of the file may be collected concurrently from multiple nodes participating in the BitTorrent network (as opposed to a central server responding with all the relevant packets in the client-server model), the BitTorrent protocol can offer advantages with respect to both the availability of a particular file and also the overall speed and resiliency of the network. This is because unlike a client-server model, file distribution is not reliant on a single point of failure or any single bandwidth bottleneck. The entire network

benefits by the number of peers that are connected on the network, as well as the number of peers that contain any specific information requested by a user.

13. In the context of BitTorrent, the term ***seeder*** is used to refer to a peer that has completed the download of a file. In contrast, the term ***leecher*** refers to a peer that has begun the download and thus may have no pieces, or only some pieces, of the complete file. The term ***swarm*** refers to the collection of seeders and leechers for a particular file or set of files. Once a leecher receives all of the pieces of the complete file, it possesses a complete reassembled instance of the original file(s) and is then designated a seeder.

14. A leecher can request different pieces of a file from multiple peers simultaneously in order to boost the chances of finding peers that can provide the pieces it wants. As the peers respond, the requesting leecher can download different pieces from different peers, at the same time. There are further optimizations that are conducted by the BitTorrent protocol at this stage, such as (i) breaking up of pieces into sub-pieces commonly referred to as ***blocks***, (ii) sending multiple queued requests for blocks to the peers, as well as (iii) organizing the pieces that are prioritized for download by the leecher. The first and the second optimizations are implemented such that multiple portions of a piece (*i.e.* blocks) can be received from different peers in parallel, thereby reducing the time to download the complete piece. Once all the pieces are received, the BitTorrent client effectively consolidates all the constituent pieces into the completed payload. Once the entire payload has been successfully downloaded, the BitTorrent client also reports to a tracker component that it has completed the download so that the tracker knows that the client is now a seeder.

15. Another aspect to understanding the BitTorrent protocol is the ***choking and unchoking*** process, which allows a leecher to select the other peers on the network with whom it may exchange pieces. This enables a leecher to maximize its own download rate by selecting particular peers (a default of no more than eight[3]) that will be "unchoked," *i.e.*, to whom pieces may be sent. On a technical level, this decision is made by each leecher by utilizing choking algorithms that periodically evaluate the other peers in the network. Leechers generally prioritize

[3] Frederiksen-Cross Rebuttal Report, ¶123.

unchoking of other peers that will maximize their own download rate i.e., unchoking the peers that have previously provided the best upload speed to this Leecher.

16. BitTorrent provides an effective and efficient protocol for the transfer of very large files over the Internet. BitTorrent is popular among varied institutions such as universities, and governmental and non-governmental organizations. For instance, NASA utilized BitTorrent for the distribution of data related to the "visible earth" project for distributing images of Earth for researchers and enthusiasts alike. Owing to its ability to distribute large files quickly, BitTorrent has also been utilized for distribution of security patches, as well as updates to enhance gaming experience for users. For instance, Blizzard Entertainment developers of the popular game "World of Warcraft," utilized BitTorrent for the distribution of patches to the game.[4] Since its inception in the early 2000s, BitTorrent has been influential and has been utilized by millions of users for varied purposes including facilitating research, distribution of open-source software, and even promotion of security through distribution of software patches.

It Is Unlikely That Meta Uploaded Pieces Containing Plaintiffs' Works During Either The Seeding or Leeching Phases

17. I previously explained in my Rebuttal Report that Plaintiffs had provided no evidence of actual distribution of any of their works, and in my opinion, it was unlikely that Meta did so via seeding. I explained that this would have been unlikely because of the numerous different practical and technical factors that would have had to have aligned in order for Meta to have seeded Plaintiffs' works to another peer on the network. As explained in my Rebuttal Report, after the completion of a torrent download, in order for Meta to have uploaded a piece containing the Plaintiffs' works to another peer on the network via seeding, all of the following conditions must have been met: **(i)** Meta must have previously and affirmatively initiated a connection with the receiving peer (as Meta's network firewall configuration would otherwise have blocked inbound connection requests from that peer),[5] **(ii)** the receiving peer must not have already downloaded the

[4] Frederiksen-Cross Rebuttal Report, *e.g.*, ¶¶ 70-71.

[5] I further discuss the holepunch extension discussed in the Choffnes Report in the sections below. *See*: Choffnes Report, ¶11.

pieces containing Plaintiffs' works from another peer, **(iii)** the receiving peer must have been requesting one or more pieces containing Plaintiffs' works (highly unlikely given the negligible percentage of pieces in the at-issue datasets that contained Plaintiffs' works), **(iv)** the receiving peer must have selected Meta for providing that piece over all of the other available peers on the network that also had that piece, and **(v)** the peer must have remained in one of Meta's unchoke slots for long enough to download a piece containing Plaintiffs' works.[6] As previously explained, a "piece" refers to a portion of the data (payload) of a particular torrent file.

18. I also explained, as an additional factor relevant to seeding, that **(vi)** the peer must also have received the piece from Meta in a time window of no more than 60 seconds, because Meta's torrent download script was specifically designed to minimize seeding by disconnecting from the BitTorrent network as soon as it discovered that the download was complete, which was within no more than 60 seconds after completion of the torrent download.[7] All of these factors combine to make it highly unlikely that any piece of any Plaintiff's work was seeded by Meta. I adhere to my prior opinions regarding seeding.[8]

19. The Choffnes Report asserts that there was a greater than 99% chance that Meta shared a piece of Plaintiffs' works with respect to ZLib and Internet Archive,[9] relying primarily on the "leeching" phase before the torrent download was complete. Nevertheless, he does not appear to directly address the factors outlined in my Rebuttal Report; he instead glosses over them using a number of unwarranted assumptions and mischaracterizations of how the BitTorrent protocol works, to arrive at a statistical probability model that grossly inflates the probability of distribution. Nevertheless, factors **(i)** through **(v)** outlined above with respect to seeding also apply to uploading during the leeching period. The following additional factors also apply:

---

[6] Frederiksen-Cross Rebuttal Report, *e.g.*, ¶¶ 17, 127-130, 132-134.

[7] Frederiksen-Cross Rebuttal Report, *e.g.*, ¶¶ 131, 98-100.

[8] In fact, I note that the Choffnes Report does not provide any opinions about the probability of Meta having uploaded a piece of Plaintiffs' works through seeding, as the probability calculations presented are based entirely on the periods of time that, he contends, Meta was still downloading/leeching. (Choffnes Report ¶¶ 21-23, Table 2.)

[9] The Choffnes Report also asserts that there was a 72.91% chance that Meta shared a piece of the Plaintiffs' works with respect to the LibGen Non-Fiction library, which I also address in this declaration.

20. **For uploading during leeching to have occurred, Meta must also have already downloaded the piece containing Plaintiffs' works**. Under the scenario addressed in my Rebuttal Report in which Meta was acting as an alleged seeder, Meta would have had ***all*** the pieces pertaining to that torrent file and would be available to share (for no more than 60 seconds). But when considering the leeching phase prior to completion of the download, by definition, Meta would have downloaded only ***some*** of the pieces of the torrent, and only those pieces would have been available for potentially uploading to other peers. Once a complete piece has been downloaded and verified, the downloading peer sends a "HAVE" message to advertise the availability of the new piece to the peers for which it already has a connection[10] – only then could other peers request that piece. As discussed in my Rebuttal Report, the order in which pieces are downloaded is non-deterministic and depends on the swarm dynamics for that specific torrent file; the timing of when Meta obtained these pieces cannot be determined retroactively. If the pieces pertaining to the Plaintiffs' works were acquired towards the end of the downloading process, for example, the window to share those pieces with others would be small.

21. **For uploading during leeching to have occurred, Meta must also have unchoked the receiving peer based on "tit-for-tat" strategy**. During the *seeding* phase after the download of the entire torrent payload for a given torrent file, libtorrent uses a round-robin method of unchoking, checking the eight unchoke slots every 15 seconds, and potentially rotating out peers. But during the *leeching* phase, libtorrent uses a "tit-for-tat" strategy for unchoking peers,[11] in which peers that have most quickly provided data to Meta are prioritized for unchoking.[12] These decisions are made dynamically based on the bandwidth provided by each peer. For instance, the peer with the fastest bandwidth, and thus placed in Meta's unchoke slots, may not have required any pieces comprising the Plaintiffs' works.

---

[10] In some cases, this information is transmitted among peers using a bitfield. *See*: Jules Sam Randolph, "Answer to 'How Does Bittorrent Work?,'" Super User, October 25, 2014, https://superuser.com/a/831673.

[11] "Libtorrent," libtorrent, accessed March 20, 2025, https://libtorrent.org/reference-Settings.html#choking_algorithm.

[12] Unless the peer was subject of an optimistic unchoke, which are limited for only 20% of all unchoke slots (20%) and only rotate on an interval of 30 seconds. *See*: "Libtorrent," libtorrent, accessed March 20, 2025, https://www.libtorrent.org/reference-Settings.html#settings_pack.

22. **For uploading during leeching to have occurred, Meta must also have uploaded sub-pieces or blocks actually containing Plaintiffs' works**. Another factor outlined in my Rebuttal Report and not disputed by the Choffnes Report is the small proportion of pieces in the torrent files that contain Plaintiffs' works.[13] For the ZLib dataset, for example, all the pieces containing Plaintiffs' works combined make up 0.023% of the torrent files that contain Plaintiff works (a subset of all downloaded torrent files).[14] Accordingly, if ZLib were represented as the 52 mile distance between San Jose and San Francisco, all the pieces of the Plaintiffs' works would be a little less than two bus lengths.[15] Similarly for the IA dataset, all the pieces of the Plaintiffs' works combined make up 0.056% of the torrent files that contain Plaintiff works.[16] Using the same analogy of the distance from San Jose to San Francisco, all of Plaintiffs' books in the IA dataset would stretch roughly over four bus lengths. As I previously explained, the fact that Plaintiffs' works constitute a negligible percentage of the overall payload reduces the likelihood that Meta's AWS instances ever seeded pieces containing Plaintiffs' works.[17]

23. The Choffnes Report's new focus on uploading during "leeching" highlights an additional problem; his report provides a probability estimate as to whether Meta uploaded ***a piece*** of Plaintiffs' works but data is ***not*** exchanged among BitTorrent peers in entire pieces but in sub-pieces commonly referred to as ***blocks***. These blocks are typically 16 kilobytes (16K) in size.[18] For example, a piece four megabytes (4MB) in size would have 256 individual blocks. These blocks thus constitute a small fraction of the overall piece, which itself constitutes a small portion of the overall torrent payload. Critically, different blocks within a piece can be obtained from multiple peers simultaneously, thus reducing the chance that any particular block would have come

---

[13] Frederiksen-Cross Rebuttal Report, ¶¶ 117-121.

[14] Frederiksen-Cross Rebuttal Report, Table 4.

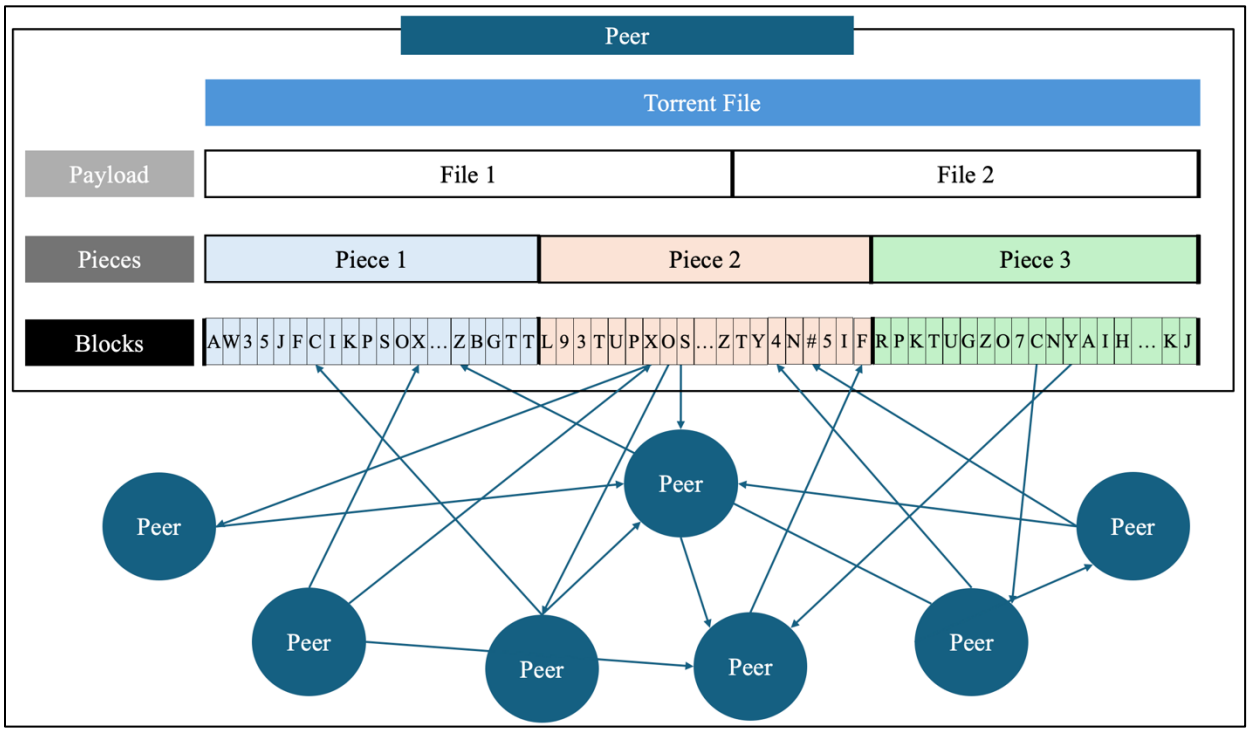
[15] Frederiksen-Cross Rebuttal Report, ¶¶ 120.

[16] Frederiksen-Cross Rebuttal Report, Table 4.

[17] As previously explained in my Rebuttal Report, I observed more than 660 copies of the Plaintiffs' works across the three downloaded datasets; this was attributable to the fact that these datasets tend to exhibit significant amounts of duplication. For example, the ZLib dataset alone had 42 copies of Laura Lippman's What the Dead Know, and 40 copies of Jacqueline Woodson's Another Brooklyn. Frederiksen-Cross Rebuttal Report, ¶ 161, Tables 9-10.

[18] Frederiksen-Cross Rebuttal Report, ¶¶ 65-66.

from a given peer (such as Meta) as opposed to some other peer on the network. The figure below shows a highly-simplified illustrative breakdown of files into pieces, and further into blocks, which are exchanged among multiple peers:[19]



24. As shown above, the upload of data within the BitTorrent protocol is in fact an upload of these sub-pieces or blocks, with different blocks provided simultaneously by multiple different peers and in a non-deterministic order. The uploading peers can also include seeders who do not require blocks from any other peers (discussed further below), and as illustrated above, it is possible for the underlying files to straddle more than one piece. After the BitTorrent client installed on the receiving peer's computer receives all of the blocks for a specific piece, it reassembles the piece. Unless all of the blocks have been received, reassembled, and verified, a particular piece remains unusable. Partially received pieces, for which all the underlying blocks have not been received, are not written to disk.[20] This means that the received blocks are retained

[19] The simplified illustrative highlights core concepts of the BitTorrent protocol. In practice, each piece is subdivided into hundreds of smaller blocks, and torrent swarms may include more peers.

[20] "Libtorrent Manual," accessed March 20, 2025, https://www.libtorrent.org/tuning.html#understanding-the-disk-threads.

in temporary storage (or a buffer) until all the blocks comprising the piece have been received.[21] If there are missing blocks for a given piece, such pieces are discarded and not written to disk.[22] Similarly, once a piece is reconstructed from its constituent blocks, it is verified against the piece hash from the .torrent file. If the hashes do not match the newly assembled piece is discarded in its entirety.[23,24] The download of pieces containing the Plaintiffs' works, in other words, is only completed after all the blocks that make up those pieces are received.

25. The lack of usability of these blocks is further compounded by the fact that the Plaintiffs' works in the downloaded datasets are not in "plain text" or "ASCII text," or some other format readily usable by a human being. They instead contain elements such as formatting metadata and other internal structures that must be parsed by software to present the content to the user in an understandable way. Compression presents another problem; for example, the popular EPUB format, the most prevalent format with respect to the files constituting Plaintiffs' works, uses ZIP data compression,[25] a commonly used compression algorithm for text data. ZIP decompression relies on a "dictionary" built up from previous data in the file to reconstruct the original file. Accordingly, a given "block" that provides a small portion of a larger compressed file will likely be unusable by itself because the computer will not have the dictionary used for decompression that was built from the previous data in the file. With respect to PDF files, the

[21] The blocks may also be written to disk, if the cache memory is full. However, these pieces still remain unusable until compiled to form a piece by the BitTorrent client. For instance, if all the blocks are not received for a PDF file, the PDF reader application may not be able to read the PDF if it is missing key components, such as the header, object definitions, etc. *See*: "Libtorrent Manual," accessed March 20, 2025, https://www.libtorrent.org/tuning.html#understanding-the-disk-threads, and "Resolve Damaged Document Error When Opening PDF Files," accessed March 20, 2025, https://web.archive.org/web/20240127090015/https:/helpx.adobe.com/acrobat/kb/pdf-error-1015-11001-update.html.

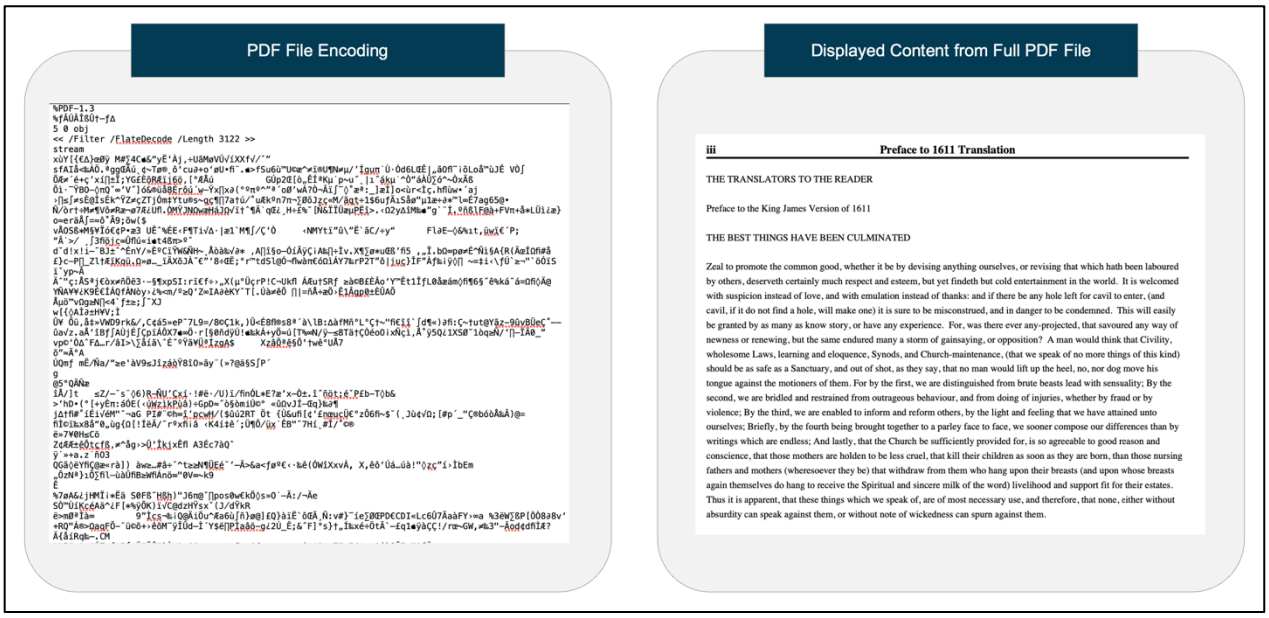
[22] "Bep_0052.Rst_post," accessed March 20, 2025, https://www.bittorrent.org/beps/bep_0052.html.

[23] "Bep_0052.Rst_post," accessed March 20, 2025, https://www.bittorrent.org/beps/bep_0052.html.

[24] BitTorrent Limited, "Help Center - What Do the Terms 'hashfails' and 'Wasted' Mean?," BitTorrent, accessed March 21, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000022802-what-do-the-terms-hashfails-and-wasted-mean-.

[25] "FlightDeck - Handbook - Zipping an EPUB," accessed March 21, 2025, https://ebookflightdeck.com/handbook/zipping.

second most prevalent format, they may also include compressed data and, if they were not subjected to character recognition, could contain little more than scanned book page images with no recognizable text strings. Accordingly, even if a receiving peer were hypothetically able to capture and view individual blocks that included some portion of the Plaintiffs' works, they would be little more than unusable gibberish before the other blocks constituting the file were downloaded. The figure below demonstrates the encoded format of an exemplary PDF files, in contrast with the content from the complete PDF file when displayed (which could only occur after it was fully downloaded).[26]



26. The Choffnes Report does not present any probability that Meta shared any ***blocks*** pertaining to the Plaintiffs' works' during the leeching phase. As explained above, Plaintiffs' works constitute a small portion of the torrented datasets. But looking at these proportions at the ***block level*** rather than the ***piece level***, the number of blocks representing Plaintiffs' works is even lower as shown in the following table:

[26] "The King James Holy Bible," n.d., https://www.holybooks.com/wp-content/uploads/2010/05/The-Holy-Bible-King-James-Version.pdf., after removal of the cover and index pages.

Dataset	Total Number of Torrent Files Downloaded Containing Plaintiffs' Works	Number of Blocks Pertaining to Plaintiffs' works	Total Number of Blocks Across Downloaded Torrent Files Containing Plaintiffs' Works	Proportion of Blocks within Downloaded Torrent Files Containing Plaintiffs' Works
LibGen Non-Fiction (Scitech only)	2	481	1,321,728	0.036%
Internet Archive (IA)	46	148033	4,173,447,168	0.0035%
Z-Library (ZLib)	146	33387	1,965,765,120	0.0017%

27. As a result, it is highly unlikely that *Meta* (as opposed to other peers on the network) would have shared any significant number of blocks from pieces constituting the Plaintiffs' works to any single peer. And even if Meta had uploaded a single block containing some portion of a Plaintiff's work to another peer, this would have amounted to a negligible amount of content relative to the size of the work that alone would itself have been unusable; the remaining blocks would likely have been obtained from other peers in the network.

28. Individual blocks are unusable until all the blocks have been received,[27] and, the chance that Meta shared all blocks comprising a piece of a Plaintiffs' works to peers during the leeching phase of the torrent download process is unlikely. As a result, even if the Choffnes Report had provided a reliable probability estimate (which as I will explain below, it did not) that Meta had shared some portion of Plaintiffs' works during the leeching process, it is unlikely that Meta uploaded all (or any usable number of) the blocks in a particular piece that contain Plaintiffs' works. Accordingly, in light of all of the constraints and conditions that must have aligned, it is highly unlikely that Meta uploaded entire usable pieces containing Plaintiffs' works to other peers during download of the at-issue datasets.

[27] Even if a specific file is less than a piece size, it would not be accessible until all the blocks comprising that piece have been received, assembled, verified, and saved, as discussed above.

29. My opinion is further supported by Amazon Web Service (AWS) cost and usage data that reports the amount of data that was uploaded and downloaded by Meta's AWS instances between March and July 2024,[28] when the download process for LibGen, ZLib, and IA took place.[29] This data (which I understand was produced in discovery in this case on March 19, 2025 pursuant to a limited reopening of discovery) indicates that, on average, the amount of data uploaded from these AWS instances to the Internet would not have exceeded approximately 30% of the amount of data that these AWS instances downloaded from the Internet in connection with the April to July 2024 torrent download for Internet Archive, ZLib, and LibGen. On average, peers could have received from Meta, approximately 30% of the data that Meta downloaded, further indicating that it was unlikely that Meta uploaded any of the Plaintiffs' works (or any usable portion of them) to peers during either the seeding or the leeching phases.

The Choffnes Report Rests on Incorrect Models and Assumptions to Arrive at an Inflated Probability Estimate

30. The Choffnes Report makes numerous assumptions in order to overestimate Meta's potential involvement in sharing any of the Plaintiffs' works. These assumptions fail to capture the complexity of the torrenting process and instead rely on unsubstantiated generalizations.

31. ***Bernoulli Trials Are Unhelpful Here***: The Choffnes report relies "on a Bernoulli experiment," "[t]o calculate the probability that Meta shared at least one piece of Plaintiffs' works." (Choffnes Report, ¶24) A Bernoulli trial provides a probability in a simplified scenario involving one or more independent events that have exactly two possible outcomes (*i.e.*, success or failure,

---

[28] The AWS billing data contained within Meta_Kadrey_00237299, was provided to me by counsel, after verifying that these billing logs pertain to instances that were utilized by Meta for the torrent downloads of LibGen, ZLib, and IA that took place in April-July 2024. Duplicate rows are removed from the data, resulting in the 30% upload to download ratio through April to July 2024.

[29] Table 3 of my Rebuttal Report reported a total download of 267.4 TB across LibGen Non-Fic, IA and ZLib in 2024. A closer inspection of the file lists for IA and ZLib revealed that they contain both the .tar files and the individual files that exist within the same .tar files. These .tar files were counted towards the calculated total download. After deduplicating by removing all .tar files, we obtain deduplicated total download amounts of 112.9 TB for IA (as opposed to 193.5 TB), and 49.17 TB for ZLib (as opposed to 63.6 TB). Therefore, the total data downloaded by Meta was 172.4TB across IA, ZLib, and LibGen Non-Fiction in 2024. This does not change the piece level percentages presented in Tables 4, 12, 13, and 14 of my Rebuttal Report.

true or false, etc.). A textbook example of a Bernoulli trial, described in the Wikipedia page cited by the Choffnes Report,[30] is flipping a coin; each flip has exactly two possible outcomes (heads or tails), so the probability of either outcome is 0.5, and that probability remains the same for each successive coin flip. But the underlying conditions that make a Bernoulli experiment potentially reliable for estimating probabilities do not exist when applied to the probability of Meta having uploaded via the BitTorrent protocol.

32. One of the most important conditions in the Bernoulli experiment is that each trial (*i.e.*, upload opportunity) be *independent* of other trials.[31] However, within the BitTorrent protocol, each upload opportunity influences the next, because the decision to choke peers is dependent on how much data is provided by the peer during the last opportunity. For instance, if the upload of data to a peer succeeds, that has an immediate effect on the bandwidth calculations conducted to select peers in the next cycle of upload.[32]

33. In order to use a Bernoulli trial, therefore, the Choffnes Report is forced to assume that "the probability of BitTorrent picking a piece of Plaintiffs' works is fixed and statistically independent." (Choffnes Report, ¶24.) But it is not, as the probability would have depended on a number of dynamic interdependencies and variables including the shifting constitution of the torrent swarm, the particular pieces that peers have already downloaded, the pieces that are requested by other peers, the bandwidth of the peers in the swarm, and a number of other interdependent factors as described below. And framing the probability as a binary question of whether Meta uploaded a "piece" containing Plaintiffs' works is overly simplified because, as noted, peers can download blocks within the same piece in parallel from multiple different peers. The question of whether Meta uploaded a "piece," as a yes/no question, is misguided and ignores the partial nature of any potential contribution.

---

[30] "Bernoulli Trial," in Wikipedia, March 16, 2025, https://en.wikipedia.org/w/index.php?title=Bernoulli_trial&oldid=1280820336 (cited in Choffnes Report, ¶24 n.12.)

[31] Alvin W Drake, "Fundamentals of Applied Probability Theory." p,124.

[32] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf. "*the value of bandwidth shifts rapidly over time as resources go away and become available,*" p,4.

34. ***Inflated Assumptions Regarding Opportunities to Upload***: The Choffnes Report also relies heavily on the idea that there were many potential opportunities for Meta to have uploaded during the leeching process, which he calculates by multiplying the number of unchoke slots by a calculated number of unchoke intervals. (Choffnes Report, ¶23, Table 2.) A key problem with this assumption is that it captures, at most, a *potential* opportunity for uploading and not an instance of *actual* uploading – let alone uploading of a piece containing Plaintiffs' works. By assuming that every unchoke interval represents an *actual* upload of data, the Choffnes Report ignores the prerequisites that must be satisfied for the upload to take place. A peer being unchoked, in other words, does not necessarily indicate that Meta actually uploaded to that peer, let alone that Meta actually uploaded any portions of any Plaintiff's book to that peer.

35. As discussed in my Rebuttal Report, setting aside any consideration of blocks, a peer must still have requested the specific piece constituting a Plaintiff's works from Meta, in order to receive that piece at the time the peer is in an unchoked slot.[33] A peer simply being unchoked by Meta does not indicate that Meta actually uploaded data to that peer,[34] and the Choffnes Report ignores important considerations such as whether the peer has interest in a specific piece containing the Plaintiffs' works (as well as the other factors identified in the previous section above).

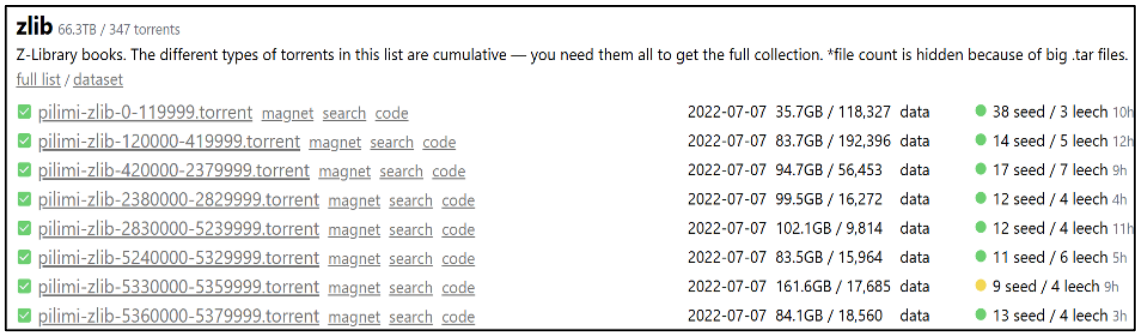
36. Another important consideration that the Choffnes Report ignores was the constitution of the swarm during torrent download process. The Choffnes Report's calculation of upload opportunities assumes that the swarm would have included at least eight (8) leechers present to occupy all of Meta's unchoke slots through the entire duration of the torrent download. But the Choffnes Report provides no evidence whatsoever to support this assumption, and as noted above, the AWS cost and usage data produced by Meta undermines this assumption.

37. And although the constituents of a swarm are dynamic and can continuously change, the list of torrents from the Annas-Archive website provides some indication of swarm composition

[33] Frederiksen-Cross Rebuttal Report, ¶¶123, 124.

[34] "*Note that this does not necessarily mean that peer B is uploading data to A, but rather that B will upload to A if A issues a data request*." Arnaud Legout et al., "Clustering and Sharing Incentives in BitTorrent Systems," https://read.seas.harvard.edu/~kohler/pubs/legout07clustering.pdf.

as it provides a current list of torrents with a number of active seeders and leechers in the swarm. For example, the following list of the first eight torrents from Annas-Archive for the ZLib dataset show fewer leechers but a significant number of seeders:[35]



zlib 66.3TB / 347 torrents

Z-Library books. The different types of torrents in this list are cumulative — you need them all to get the full collection. *file count is hidden because of big .tar files.

full list / dataset

pilimi-zlib-0-119999.torrent	magnet search code	2022-07-07	35.7GB / 118,327	data	38 seed / 3 leech 10h
pilimi-zlib-120000-419999.torrent	magnet search code	2022-07-07	83.7GB / 192,396	data	14 seed / 5 leech 12h
pilimi-zlib-420000-2379999.torrent	magnet search code	2022-07-07	94.7GB / 56,453	data	17 seed / 7 leech 9h
pilimi-zlib-2380000-2829999.torrent	magnet search code	2022-07-07	99.5GB / 16,272	data	12 seed / 4 leech 4h
pilimi-zlib-2830000-5239999.torrent	magnet search code	2022-07-07	102.1GB / 9,814	data	12 seed / 4 leech 11h
pilimi-zlib-5240000-5329999.torrent	magnet search code	2022-07-07	83.5GB / 15,964	data	11 seed / 6 leech 5h
pilimi-zlib-5330000-5359999.torrent	magnet search code	2022-07-07	161.6GB / 17,685	data	9 seed / 4 leech 9h
pilimi-zlib-5360000-5379999.torrent	magnet search code	2022-07-07	84.1GB / 18,560	data	13 seed / 4 leech 3h

38. In this example, had Meta been involved in any of the swarms listed above, it would not have filled eight unchoke slots that are a key assumption behind the probability estimates in the Choffnes Report. And even if other peers had been assigned to Meta's unchoke slots, the Choffnes Report does not consider the presence of the ***seeders*** in the swarm who already downloaded the entire torrent file, who would have provided attractive alternatives in providing blocks as compared to a non-seeding peer such as Meta. The Choffnes Report does not provide any evidence or even reasoned analysis to support the assumption that eight (8) leechers would have been present in the swarm (in addition to Meta), occupying all eight unchoke slots, through the entire duration of the download process.

39. The fact that the Choffnes Report vastly overestimates the amount of data that was likely uploaded by Meta is further confirmed by the fact that, as noted in Paragraph 29 above, Meta could only have uploaded a maximum of approximately 30% of the data it downloaded (based on the cost and usage data discussed above), during the download process that took place between April-July 2024. This data further undermines the assumption that the number of upload "opportunities" reported in the Choffnes Report represent actual uploads. By relying on calculations based only on unchoke intervals and unchoke slots, the Choffnes Report derives a hypothetical estimate for the number of times that Meta's libtorrent client "had an opportunity to offer a new piece to a new peer[.]" (Choffnes Report, ¶21.) But as the table below shows, this

[35] "Torrents ZLib - Anna's Archive," accessed March 20, 2025, https://annas-archive.org/torrents/zlib.

number of opportunities "to offer a new piece to a new peer" results in an upload-to-download ratio that cannot be reconciled with the actual data:

Dataset	Total Number of Pieces in Torrents Containing Plaintiffs' Works	Choffnes Report Upload Opportunities	Calculated Upload To Download Ratio Utilizing Methodology in the Choffnes Report
LibGen Non-Fiction	5,163	3,840	74.38%
IA	353,119	507,840	143.82%
ZLib	1,468,494	5,115,840	348.37%

40. The second column shows the total number of pieces in the particular downloaded torrents that contain Plaintiffs' works, and the third column shows the number of upload opportunities from the Choffnes Report for those works (calculated by multiplying the values in Columns 3 and 4 in Table 2). (Choffnes Report, ¶23, Table 2.) As shown, the number of upload opportunities presented by the Choffnes Report presume that Meta uploaded more data than it actually did; by assuming that all upload opportunities resulted in actual uploads, his calculations result in upload-to-download ratios that far exceed the actual data – by roughly 3x for LibGen Non-Fiction, by roughly 5x for IA, and by roughly 12x for ZLib. In other words, the fact that Meta could only have uploaded a maximum of approximately 30% of the data it downloaded indicates that the Choffnes Report's assumptions about upload opportunities and actual uploading are vastly overstated.

41. ***Inflated Assumptions Regarding Availability of Pieces from Meta***: The probabilities presented in the Choffnes Report further rely on the chance of "mutual interest in one piece" by a peer and Meta. This "mutual interest" relies on two assumptions that consider that at a particular point in time, (i) the chance that Meta has a piece containing the Plaintiffs' work already downloaded and ready to share, is 50%, and (ii) the chance that Meta unchoked the peer is 50%. (Choffnes Report, ¶21.) Both of these assumptions are gross oversimplifications that lack any evidentiary support and fail to capture the complexity of the dynamic process of downloading via the BitTorrent protocol and ignores key characteristics such as the receiving peer's "interest" state at the time it is unchoked.

42. As discussed in my Rebuttal Report, many BitTorrent clients rely on a rarest-first approach, which prioritizes downloading pieces that fewer peers currently possess.[36] This ensures that otherwise-scarce pieces are disseminated throughout the swarm, but it also means that truly rare pieces may be harder to obtain early on. As a result, whether Meta's AWS instances downloaded—and therefore could have shared—a particular piece containing the Plaintiffs' works would have depended, in large part, on how available that piece was among the swarm at the time of download. Consequently, the chance that Meta held a piece containing Plaintiffs' works that was available to be shared with other peers could not be arbitrarily and retroactively assumed to be a discrete value (as the Choffnes Report assumes), because the probability of obtaining a piece with the Plaintiffs' works would have been influenced by both its relative rarity and the dynamic behavior of the swarm.

43. Additionally, the Choffnes Report considers "a 50% chance that the peer connected to Meta's libtorrent client has a piece that Meta wants. In this case, the peer is unchoked due to mutual interest." (Choffnes Report, ¶21.) But as discussed above, the decisions for unchoking a peer at a particular time are not dictated solely by "mutual interest," but instead made after consideration of other factors such as the tit-for-tat strategy, which considers the speed at which pieces are uploaded by another peer to Meta. Therefore, the chance that a particular peer is unchoked depends on each peer's individual bandwidth and whether it has previously uploaded any data to Meta. In turn, this metric also depends on the number of peers in a swarm, and the health of the swarm (i.e., if there are more seeders or leechers). Measuring such metrics for each peer as a single discrete value, as done in the Choffnes Report, will inevitably lead to an overestimate of uploading data by any one peer. Thus, the stated probability for unchoking a given peer is incorrect in the Choffnes Report.

44. ***Unwarranted Assumption that Meta Uploaded an Entire Piece Containing Plaintiffs' Works***: As discussed above, the Choffnes Report also analyzes probabilities entirely at the level of an ***entire*** piece and fails to account for its many sub-pieces or blocks.

[36] Frederiksen-Cross Rebuttal Report, ¶¶ 65, 125.

45. ***Unwarranted Discounting of Meta's Network Firewall Configuration***: The Choffnes Report also attempts to brush off the significance of Meta's firewall protections that would have blocked inbound connections unless Meta had previously and affirmatively initiated a connection with the receiving peer. (Choffnes Report, ¶¶8-11.) The Choffnes Report states that every home router firewall shares with Meta this "standard" behavior of blocking unsolicited inbound connections. This fails to recognize that instead of a single standard there are multiple ways firewalls can work. The effect of different firewall setups is explained in the stackoverflow.com post referenced by the Choffnes Report: "Also, if the [sic] neither NAT is a full-cone (or let's say, p2p-friendly) it may not be possible for the peers to connect. A p2p-friendly NAT generally accepts incoming connections from IPs they have not had any interaction with previously." (Choffnes Report ¶11 footnote 3.) As demonstrated by this citation, while it is true that the BitTorrent protocol provides a "holepunch" feature, this mechanism is not always supported. Moreover, even when enabled, there is no guarantee that the requested connection will be successful. For example, in order to use the "holepunch" feature to communicate with a peer behind the firewall, the initial peer requires the assistance of a "relay" peer. However, if the initial peer selects a relay peer that is not already connected to the peer behind the firewall, the holepunch fails. This situation happens often because the holepunch extension does not provide a way for the original peer to learn which relay peers are connected to their peer of interest. The same stackoverflow.com post referenced by the Choffnes Report also states about holepunch that "[t]his only works if the swarm has at least one peer that's not behind a NAT." As I discuss below, the supposed "hole-punching" method identified in the Choffnes Report is only an extension of the BitTorrent protocol and does not guarantee a successful connection. The hole-punch attempt must meet multiple conditions in order to facilitate any connection.[37] As the "Holepunch extension" page cited by the Choffnes Report states: "The initiating peer sends a rendezvous message to the relaying peer, containing the endpoint (IP address and port) of the target peer. If the relaying peer is connected to the target peer, and the target peer supports this extension, the relaying peer sends

[37] *E.g.*, the holepunch extension must be enabled for the initiating peer, the target peer, and the relay peer, and the relay peer must already be connected to the requested target peer.

a connect message to both the initiating peer and the target peer, each containing the endpoint of the other. Upon receiving the connect message, each peer is directed to initiate a uTP connection to the other peer."[38] (Choffnes Report, ¶11 footnote 4.) Accordingly, even if Meta had utilized this functionality, it would still need to send an outbound request to the relaying peer with the intention of connecting with another "target" peer. Similarly, if another initiating peer wanted to connect to Meta (now the target peer), without Meta sending the initial request, the description of "hole punch" that the Choffnes report relies on suggests that there would still need to be an outbound request from Meta's instance to initiate the uTP connection to that initiating peer. Indeed, the Choffnes Report concedes that this peer exchange protocol only "allow incoming connections by first attempting an outbound connection." (Choffnes Report, ¶11.) Therefore, despite this functionality within BitTorrent, it remains the case that Meta must have initiated the connection with any leecher with which it can exchange data, as all other leechers attempting to initiate requests for pieces of the dataset from Meta would have been rejected by Meta's network configuration.

46. Attached hereto as Exhibit A is a true and correct copy of my current curriculum vitae.

I declare under penalty of perjury that the foregoing is true and correct. Executed on this 24th day of March, in Eugene, Oregon.

Barbara Frederiksen-Cross

[38] "Bep_0055.Rst_post," accessed March 20, 2025, https://www.bittorrent.org/beps/bep_0055.html.